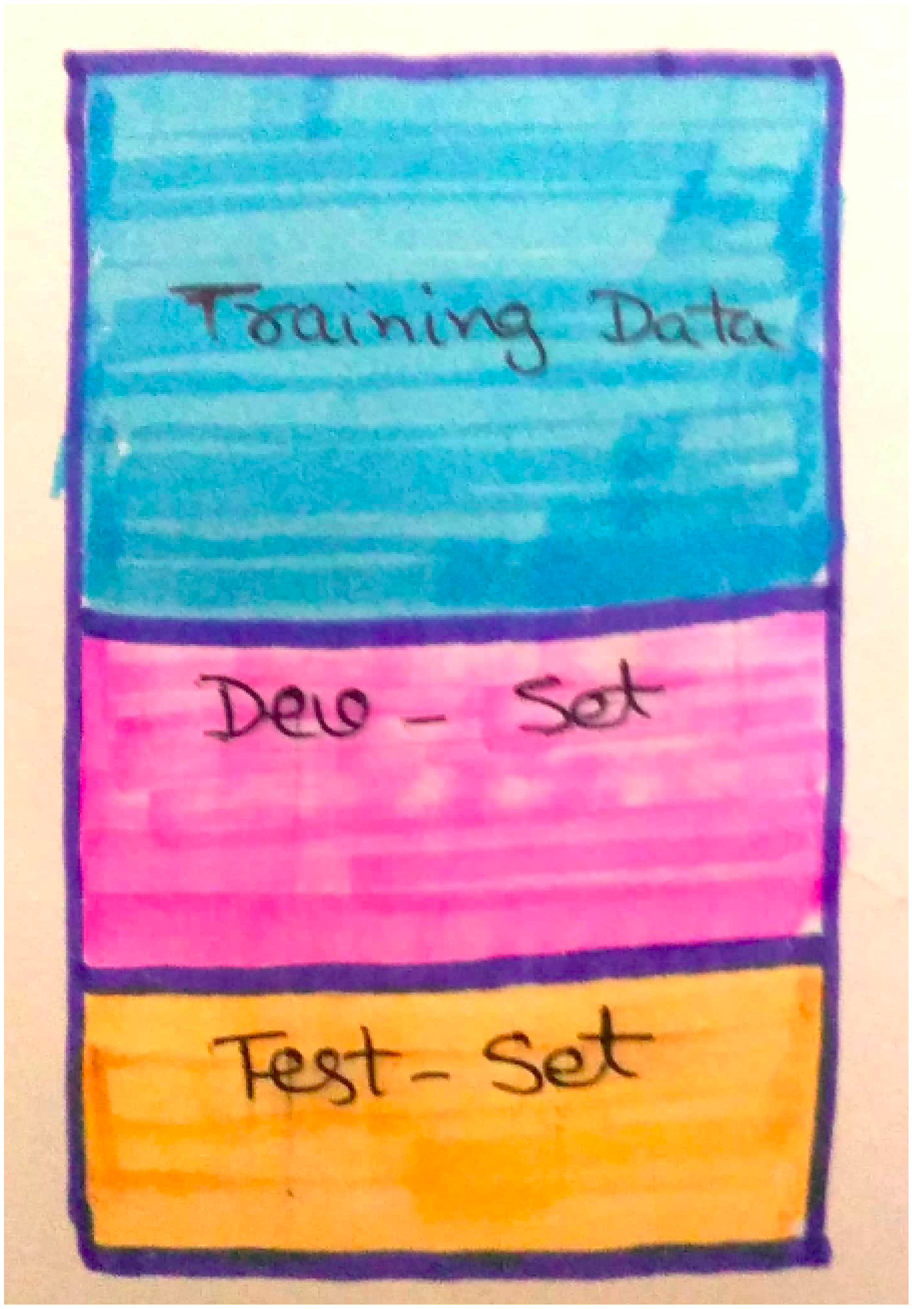
Everything You Need To Know About Train/Dev/Test Split — What, How and Why

Sanjeev Kumar Follow
Mar 17, 2019 · 6 min read



We'll try to understand **what** is train/dev/test split? **How** to split your data into train, dev, and test-set and **why** do we even need 3 different sets of data to train our machine learning models. In the end, we'll wrap up with the general advice while making decisions involving data splits.

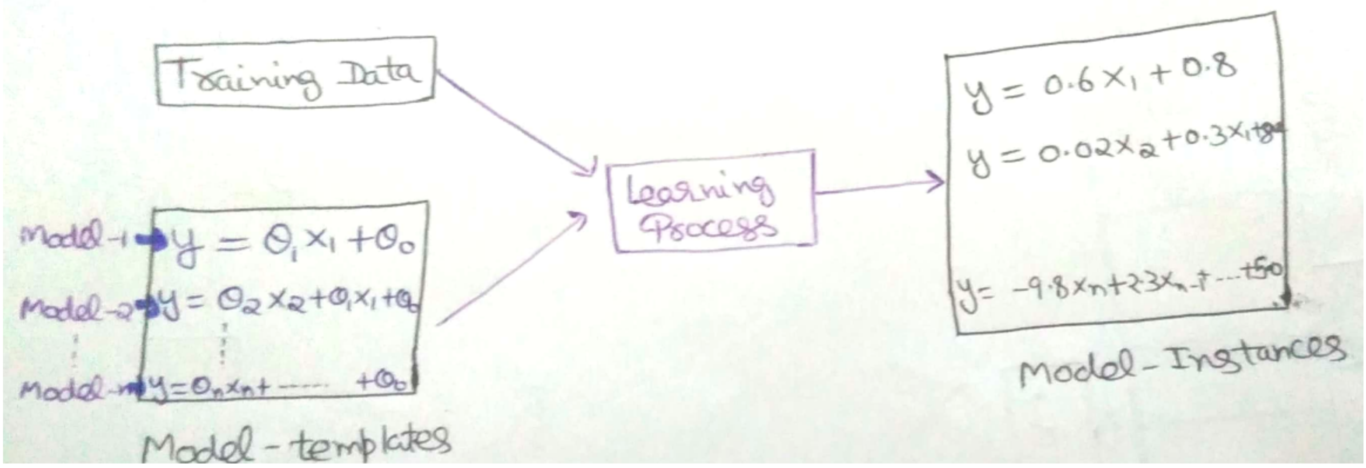
What is train/dev/test split

Training Data Learning algorithm like gradient descent use training data iteratively to learn the parameters of the model.

In the training process, data is slowly memorized into the parametric aspect of the model with the goal of generalizing this model to unseen data.

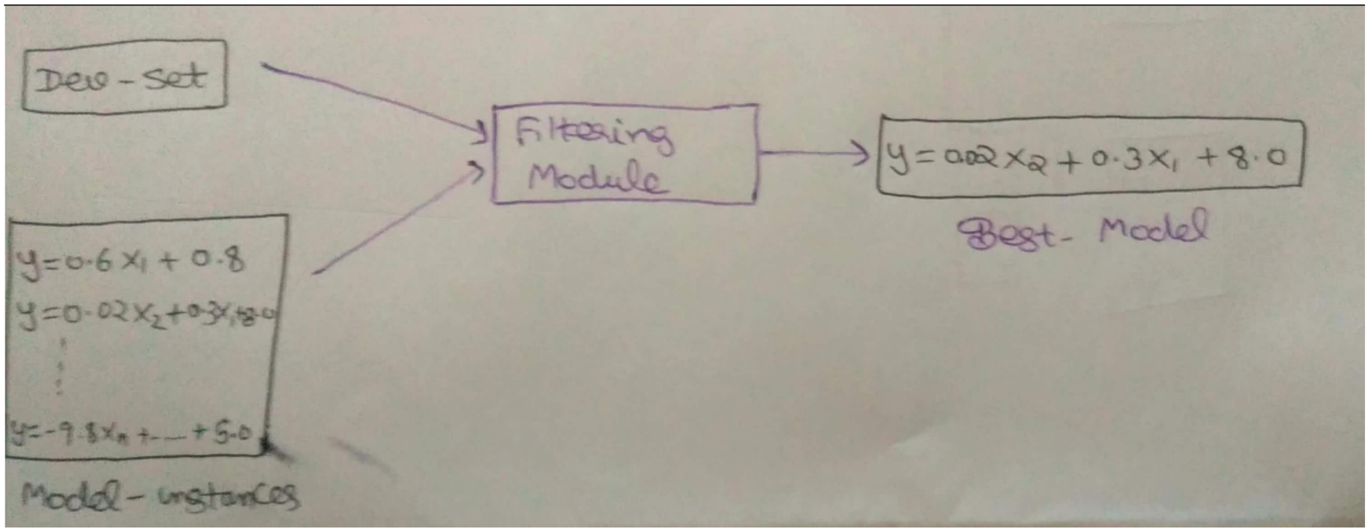
Training process emits the parameters of a model and hence the sole purpose of training data is to make a decision about which parameters to pick given huge options to choose from.

To solve any machine learning problem we have more than one ideas in our mind to try e.g. different model architectures, add regularization or not etc. We use training data to emit parameters for each of the model i.e. If we have 3 models in mind to try, we use training data to give parameters of each of the model choice but now we need to decide which model is good one out of all these choices.


Training Data
Learning Process
Model-1 → $y = \theta_1 x_1 + \theta_0$
Model-2 → $y = \theta_2 x_2 + \theta_1 x_1 + \theta_0$
Model-n → $y = \theta_n x_n + \cdots + \theta_0$
Model - templates
$y = 0.6 x_1 + 0.8$
$y = 0.02 x_2 + 0.3 x_1 + 8$
$y = -9.8 x_n + 2.3 x_{n-1} + \cdots + 50$
Model - Instances

Training Data helping learning process to instantiate models

The goal of **dev-set** is to rank the models in term of their accuracy and helps us decide which model to proceed further with. Using Dev set we rank all our models in terms of their accuracy and pick the best performing model. i.e. dev set ranks models similar to a search engine like google rank pages and then pick the top model and hence act as a filter to remove bad models.

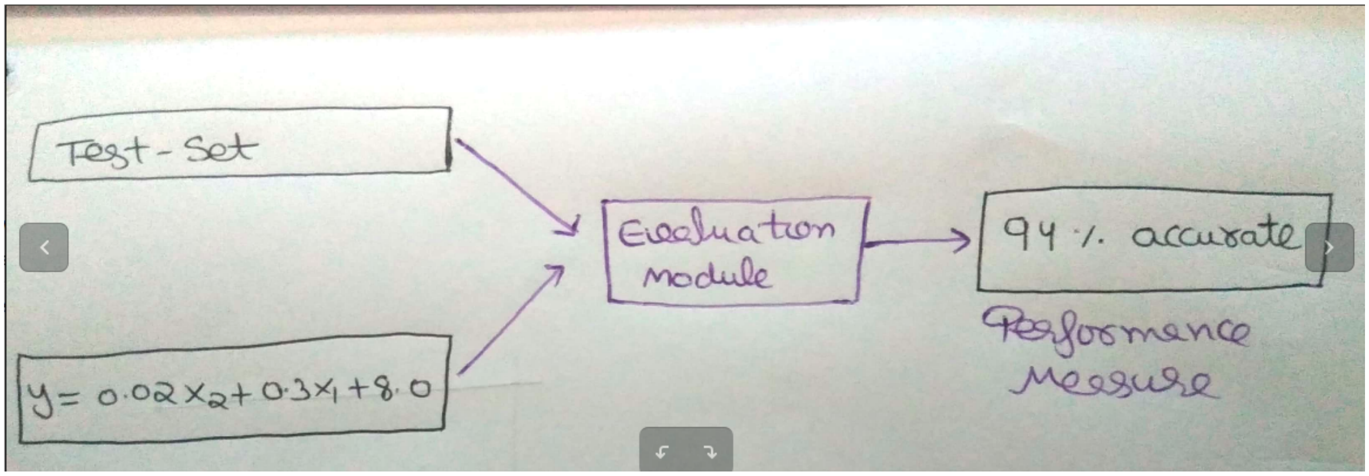


Dev-set helping filtering module to pick best model out of available choices

In the process of building a machine learning project, we made a lot of decisions from parameters of the model to the different choice of models.

Choices of parameters are made by learning algorithm and choice of model is made by us by intentionally picking the best model performing well on dev-set.

So when you have successfully consumed train and dev set while building a machine learning system you are left with the final best model out of all the ideas you tried.

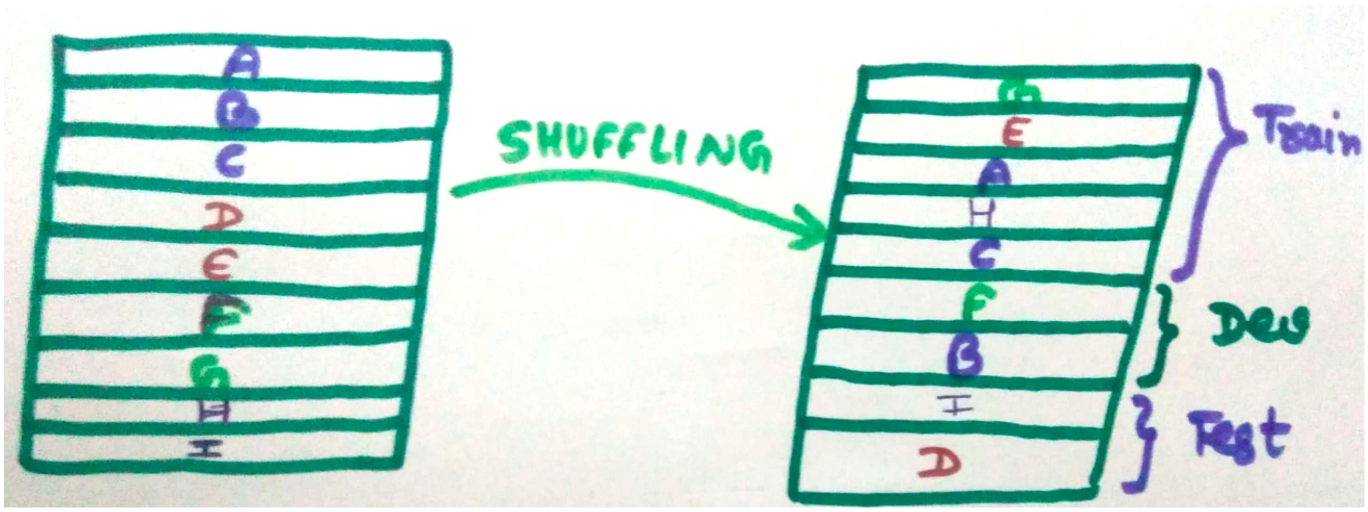


Use of test set to report the performance of the final model as a proxy for unseen data

Once the final model is available we need to be confident about the fact if this is a good model or not or how much accuracy we can expect it once we deploy it in the real world. We use **test-set** as a proxy for unseen data and evaluate our model on test-set.

How to do train/dev/test split

Deep learning systems have a huge hunger for data and the first step before even training your models is to set up your goal by creating dev/test set and optimizing metric on which to evaluate the model. An ideal way to split data is first to reorder your data by shuffling the data in random order.



How to do train/dev/test split

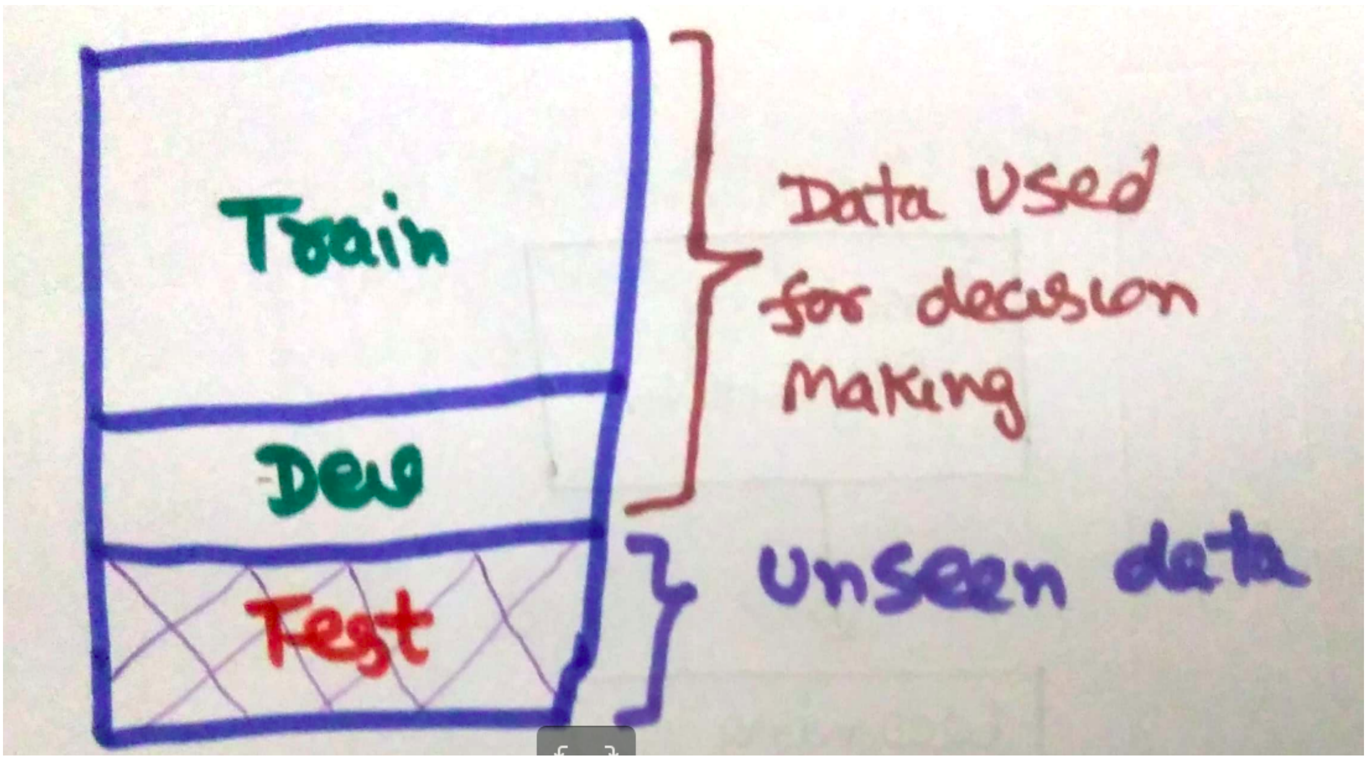
Now decide the size of your training/dev/test split. If we have a very small amount of data at our disposal we'll try to use as much data as possible for training and maybe can do the split of order 70/20/10.

If we have huge data on which to train our model then we can do the split keeping the following 2 ideas in mind:

1. Size of dev set should be large enough to pick the best model out of all the options available. Using 1000–10000 examples is the ideal number to do validation after training is done.
2. Size of test set should be large enough to give you an unbiased estimate of your model performance on unseen data and try to keep your test set around 100–10000 examples for that goal.

Why do we need train/dev/test split

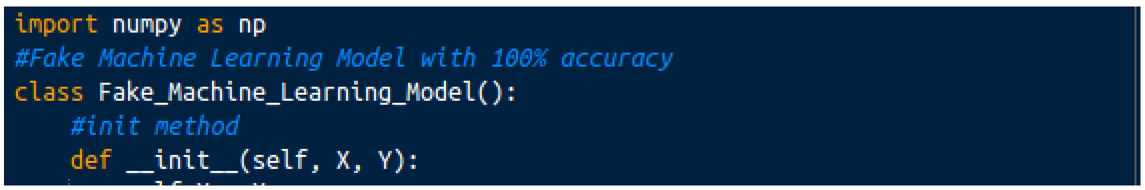
Well, you might be thinking if our goal is to train a learning system which are data hungry, why do we have to waste our data in dev and test set. Why not use the training data to report the numbers killing the need for test set?



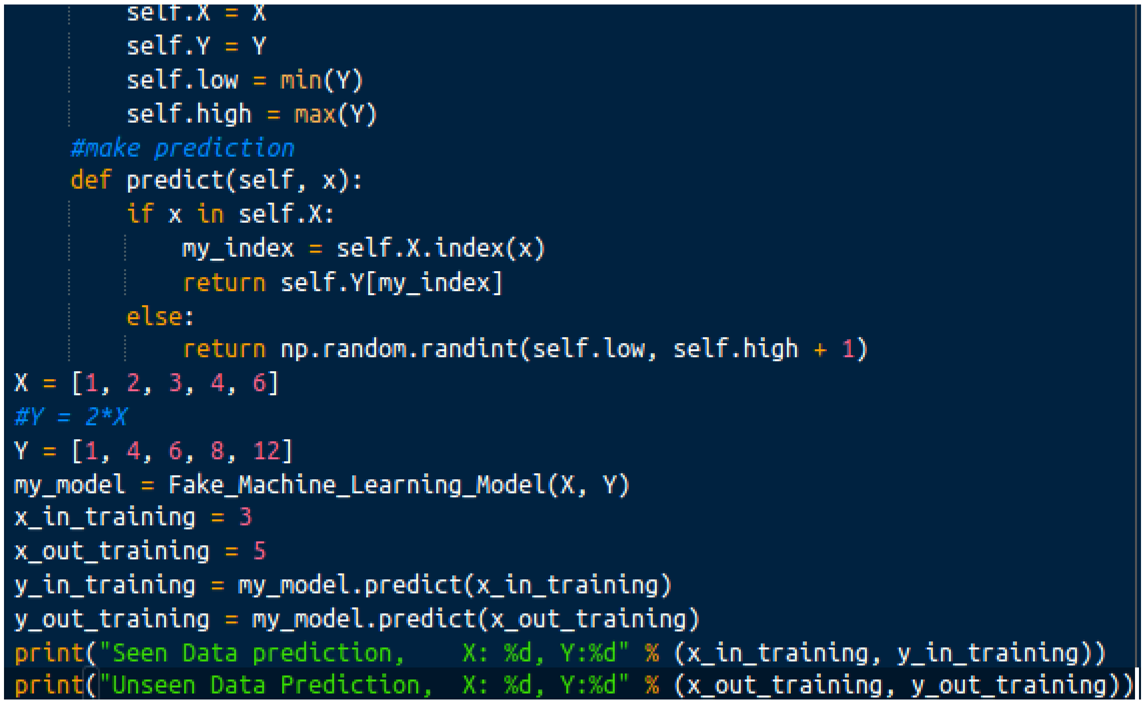
Let's try to understand the **need for test-set** by taking an example.

So learning algorithms use training data with the objective of performing really well on this training data but our goal is a generalization to unseen data than performing well on training data. If we supply entire training data to learning algorithm and then report accuracy to say 98% on same data, what we are saying indirectly is that we have built a model which can give 98% accuracy on given data but we are not sure about the unseen data.

Here's an example model for fun which gives 100% accuracy on given data but performs poorly on unseen data.



```
import numpy as np
#Fake Machine Learning Model with 100% accuracy
class Fake_Machine_Learning_Model():
    #init method
    def __init__(self, X, Y):
```



```
        self.X = X
        self.Y = Y
        self.low = min(Y)
        self.high = max(Y)
    #make prediction
    def predict(self, x):
        if x in self.X:
            my_index = self.X.index(x)
            return self.Y[my_index]
        else:
            return np.random.randint(self.low, self.high + 1)
X = [1, 2, 3, 4, 6]
#Y = 2*X
Y = [1, 4, 6, 8, 12]
my_model = Fake_Machine_Learning_Model(X, Y)
x_in_training = 3
x_out_training = 5
y_in_training = my_model.predict(x_in_training)
y_out_training = my_model.predict(x_out_training)
print("Seen Data prediction,    X: %d, Y:%d" % (x_in_training, y_in_training))
print("Unseen Data Prediction,  X: %d, Y:%d" % (x_out_training, y_out_training))
```

Fake Machine learning Model

Following the last example we now understand the need for unseen/test data while reporting the performance of the model, but **why do we need dev-set** ?

In the process of building models that learn from data, we need to find the best parameters of the model and best model out of all other available ones. If we don't have dev data then we'll train all the models and pick the model with the best performance on training data. By doing so, we are taking 2 decisions with a single process i.e.

1. Parameter choice
2. Model choice

While having dev set split, first training algorithm makes the choice for optimal parameters and then those parameters are used on dev data to help us find best model architecture as compared to both choices made together by learning algorithm itself.

Dev set helps us in reducing the complexity of diagnosis if things won't go fine i.e. we'll be able to assign error to either choice of parameters or picking up model architecture very concretely.

Lack of dev set and using only the training set doesn't give you clue about which choice went wrong and luck rather than skill will be helpful to debug your learning algorithm there and to make decision further to improve the model accuracy.

General piece of advice to avoid pitfalls:

1. Choose dev/test from the distribution that reflect what data you expect to see in the future. If the dev/test set is made up of distribution different than real data distribution, your model might not work as expected.
2. You can have dev/test distribution different from training data distribution but then your diagnosis will become heavier because you will be dealing with train/test data mismatch problem as well.
3. Choose dev and test set from same distribution because it will help you with clear diagnosis when the model will not be working on the test set, which is you have overfitted the dev set and acquire more dev data.
4. Dev set should be large enough to rank the models and test data should be large enough to give you unbiased performance measures of your model.
5. You never make any decision in your learning process by looking at the test set. In fact, you even forget that you have any more data apart from train and dev set. Once all the decisions about learning algorithm are made you use test-set to only evaluate the performance of your model.

References

Machine learning Yearning by Andrew Ng

Machine Learning Neural Networks Deep Learning Machine Learning Yearning

Data Science Advice

About Help Legal

Get the Medium app